A MULTIPLE MODEL FILTER WITHOUT MARKOV SWITCHING

March 1998

T. R. Rice
Systems Research and Technology Department
Naval Surface Warfare Center Dahlgren Division
Dahlgren, Virginia 22448-5100

A. T. Alouani
Department of ECE
Tennessee Technological University
Cookeville, Tennessee 38505

ABSTRACT

When tracking with multi-sensor systems, the set of sensors used may be asynchronous and there may be communications delays between sensor platforms and the fusion center. Despite these conditions, it is desirable that each sensor maintains an accurate track. It has been recognized for some time that the use of a multiple model filter is superior to the use of a single model filter for tracking maneuvering targets. However, existing multiple model tracking algorithms use Markov switching, assuming that the likelihoods of the target state switching between kinematic models are known. The objectives of this paper are twofold. First, it will present a Multiple Model (MM) tracking algorithm, called the ARMM algorithm, that does not assume *a priori* knowledge of the target transition probability matrix. This work attempts to relax some of the assumptions found in the most widely used MM tracking algorithm. Second, it will be shown that the ARMM algorithm can also be used as the second, and final, stage in a logical process for fusing asynchronous tracks from multiple sensors that use different kinematic models in their individual track filters.

1.0 INTRODUCTION

Assume the existence of an algorithm that can optimally fuse asynchronous track data from multiple sensors using filters with different target motion models. An optimal, Multiple Model (MM) filter would be a special case of this algorithm and could be obtained by: (1) requiring the measurement provided to each filter to be the same, and (2) assuming that the measurements are taken at the same time (synchronously). From this realization, the authors were led to consider an analytic approach used to derive two, asynchronous, track fusion algorithms for multiple sensors, whose filters use the same model, as the approach for deriving a MM filter. This resulting algorithm, which will be called the ARMM filter, uses only the filter residuals to provide an on-line capability for weighting the output of the two local tracks. This algorithm does not require *a priori* knowledge of the target state transition likelihoods. Additionally, it will be demonstrated how tracks of the same target from different sensors and generated by filters using different kinematic models (it is possible that the individual tracks can each be from a different sensor or, in other instances, that some of the sensors are producing more than one track from filters with different target models) can be combined to produce a fused system track using the ARMM. This is accomplished by applying either of the track fusion algorithms multiple times and then combining the results of the individual applications with the ARMM. This paper provides building blocks to be used

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (FROM - TO)
01-03-1998	Conference Proceedings	xx-xx-1998 to xx-xx-1998

4. TITLE AND SUBTITLE
A Multiple Model Filter Without Markov Switching
Unclassified

5a. CONTRACT NUMBER
5b. GRANT NUMBER
5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Rice, T. R. ;
Alouani, A. T. ;

5d. PROJECT NUMBER
5e. TASK NUMBER
5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME AND ADDRESS
Systems Research and Technology Department
Naval Surface Warfare Center Dahlgren Division
Dahlgren, VA22448-5100

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME AND ADDRESS
Director, CECOM RDEC
Night Vision and Electronic Sensors Directorate, Security Team
10221 Burbeck Road
Ft. Belvoir, VA22060-5806

10. SPONSOR/MONITOR'S ACRONYM(S)
11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APUBLIC RELEASE
,

13. SUPPLEMENTARY NOTES
See Also ADM201041, 1998 IRIS Proceedings on CD-ROM.

14. ABSTRACT
When tracking with multi-sensor systems, the set of sensors used may be asynchronous and there may be communications delays between sensor platforms and the fusion center. Despite these conditions, it is desirable that each sensor maintains an accurate track. It has been recognized for some time that the use of a multiple model filter is superior to the use of a single model filter for tracking maneuvering targets. However, existing multiple model tracking algorithms use Markov switching, assuming that the likelihoods of the target state switching between kinematic models are known. The objectives of this paper are twofold. First, it will present a Multiple Model (MM) tracking algorithm, called the ARMM algorithm, that does not assume a priori knowledge of the target transition probability matrix. This work attempts to relax some of the assumptions found in the most widely used MM tracking algorithm. Second, it will be shown that the ARMM algorithm can also be used as the second, and final, stage in a logical process for fusing asynchronous tracks from multiple sensors that use different kinematic models in their individual track filters.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	Public Release	20	Fenster, Lynn lfenster@dtic.mil
Unclassified	Unclassified	Unclassified			19b. TELEPHONE NUMBER International Area Code Area Code Telephone Number 703767-9007 DSN 427-9007

in the design of track fusion systems, where the tracks are provided by different sensors. These tracks may be asynchronous, have different communications delays, and be generated using different models of the same target.

This paper is organized as follows. In section 2, background on target tracking and track fusion, as it relates to MM algorithms, is presented. Next, the analytical approach is presented in section 3. Two track fusion algorithms derived using the analytical approach are presented in section 4. In section 5, the derivation of the ARMM algorithm is presented, while in section 6, simulation results obtained using the ARMM to track a simulated target are given. A summary and conclusions are given in section 7.

2.0 BACKGROUND

2.1 TARGET TRACKING

For many target tracking applications, it is not possible to predict the beginning and end of a target maneuver. Hence, when tracking a target with unknown and variable dynamics, it is difficult to decide on the dynamics model to be used by the tracking filter during a given time segment of its trajectory. Existing approaches can be divided into two classes: the Single Model (SM) approach and the Multiple Model (MM) approach.

In the SM approach only a single model filter is used at a given time (see [1]-[4], among others). The tracking starts with a filter that uses a Constant Velocity (CV) model and a low process noise until a maneuver is detected. Upon maneuver detection, the state process noise of the filter is increased in [1], while [2] estimates the target acceleration and treats it as an input to the filter. In [3], the filtered state is augmented during a maneuver, while in [4] the target acceleration is treated as a bias whose estimate is used to correct the output of the CV filter. The success of the single model approach is heavily dependent on the quality of the maneuver detector used.

The MM approach employs multiple filters at any given time, each with a single different model. The outputs of these filters are then weighted to obtain the final target track. Probability techniques are used to determine the weights to be assigned to these tracks. Since a number of models (filters) are used in parallel, contrary to the SM approach, there is no hard switching between models. This approach eliminates the need for maneuver detection and filter initialization. It is worth noting that, conceptually, this MM approach can lead to an optimal solution to the tracking problem. However, the optimal solution will become intractable due to the exponential growth with time of the number of track histories. For this reason sub-optimal solutions are considered (see [5] and [6]), where a limited number of filters are used. For example, the filter used in [6], known as the Interacting Multiple Model (IMM) filter, uses only two models: a CV and a CA (Constant Acceleration) model.

One of the shortcomings associated with existing MM approaches is that the knowledge of the probability that governs the likelihood of transitions between dynamic models (called switching coefficients) is assumed known. This assumption requires *a priori* knowledge about the target's trajectory and is clearly not reasonable when it comes to tracking unknown targets in real time. As a result, it is necessary to use offline techniques, such as running multiple Monte Carlo experiments using different trajectories and a number of "candidate" switching coefficients, to determine the "best" switching coefficients. Another, perhaps mild, deficiency of the IMM is it's use of the approximation that a sum of Gaussian probability functions (pdfs) is itself a Gaussian pdf

2.2 TRACK FUSION

In the past, track fusion algorithms have not been considered as the primary means of fusing multi-sensor data for several reasons. First, fusion algorithms that could be shown to optimally combine tracks

from sensors that reported their tracks asynchronously (either because a target's position was measured at different times or because of communications delays or both) did not exist. This reason has recently been negated by [7] for the case where all of the sensors use track filters having the same dynamical model. In order to obtain a globally optimal solution for this case, it is necessary to feed the fused track back to each of the sensors to be used as the input state [7]. Of course, this still leaves to be solved the case where there are multiple subsets of track filters with each subset using a different dynamical model in it's filters.

Second, single model filters, such as the Kalman filter, do not realistically reflect the true covariance of the track except when their dynamical model matches the dynamics of the target. Otherwise, the covariance provided by the filter is smaller than the true covariance. Since most known track fusion algorithms utilize the inverse of the covariance of the track to weight the contribution of its state vector to the fused state vector, filters that are not doing well because of a model mismatch are contributing more than they should to the fused track.

If one decides to use the SM approach to alleviate this problem, there are still grave shortcomings. SM approaches can quickly discern when a target starts maneuvering, however, they cannot accurately determine when the target has stopped maneuvering and returned to straight and level motion and will continue to track with the maneuver model filter. When this occurs, the SM model will not provide a good estimate of the covariance. Also, when the SM model transitions from one model to another, one must wait for the new filter to settle before the covariance can be considered meaningful.

On the other hand, the MM approach provides a soft switching mechanism between models that precludes the need for filter initialization and settling. Also, the ability of MM algorithms to detect a maneuver and respond to that maneuver by increasing the weight given to the filter with the more accurate target motion model is good. Therefore, MM algorithms seem to be the best candidate for filtering data from multiple sensors when the tracks from these sensors are to be fused. However, a logical approach to fusing MM algorithms is needed.

3.0 ANALYTICAL APPROACH

Begin by considering the following generic system. There exist a target being observed by one or two sensors whose dynamics of motion are assumed to be governed by

$$\dot{X}(t) = AX(t) + G\overline{W}(t), \tag{1}$$

where $X(t)$ is the target state and $\overline{W}(t)$ is a zero mean Gaussian process with covariance

$$E[\overline{W}(t)\overline{W}'(\tau)] = q(t)\delta(t-\tau). \tag{2}$$

There are two different paths that can be taken from this point.

3.1 INITIAL ASSUMPTIONS FOR THE FIRST PATH

The first path, which will be denoted as the MSSM (Multi-Sensor, Single Model) process, assumes that there are two sensors, each with a single Kalman filter to produce local tracks and each using the same dynamical model of the target. Also, it is assumed that, in general, the time at which each sensor takes a measurement of the target's position is different, in other words the sensors are asynchronous, $t_{k-1} < t_i < t_k, i = 1, 2$ (see Figure 1). The measurement model for the ith filter, i=1, 2, is given by

$$z_i(t_i) = H_i X(t_i) + V_i(t_i), \tag{3}$$

where $V_i(t_k)$ is a zero mean, white, Gaussian, measurement noise process. Here

$$X(t_{k+1}) = \Phi(t_{k+1}, t_k) X(t_k) + W(t_k), \tag{4}$$

$$\Phi(t_{k+1}, t_k) = e^{A(t_{k+1} - t_k)}, \tag{5}$$

$$W(t_k) = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau) G \overline{W}(\tau)\, d\tau, \tag{6}$$

$$E[W(t_k) W'(t_j)] = Q(t_k) \delta_{kj}, \tag{7}$$

$$E[V_i(t_k) V_i'(t_j)] = R_i(t_k) \delta_{kj}, \tag{8}$$

$$Q(t_k) = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau) G q(\tau) G' \Phi'(t_{k+1}, \tau)\, d\tau. \tag{9}$$

and the prime is used to denote matrix transpose (See [7] or [8]).

3.2 INITIAL ASSUMPTIONS FOR THE SECOND PATH

The second path, denoted as the SSMM (Single Sensor, Multi-Model) process, assumes that there is only one sensor and two Kalman filters are processing it's measurements. Therefore the tracks produced by these sensors are synchronous. However, it is assumed that one filter has a CV model (call this LP1) while the other has a CA model (call this LP2), which leads to two target motion models given by:

LP1: $$X_1(t_{k+1}) = \Phi_1(t_{k+1}, t_k) X_1(t_k) + G_1 W_1(t_k), \tag{10}$$

LP2: $$X_2(t_{k+1}) = \Phi_2(t_{k+1}, t_k) X_2(t_k) + G_2 W_2(t_k), \tag{11}$$

where

$$W_1(t_k) \sim N(0, q_1), \tag{12}$$

$$W_2(t_k) \sim N(0, q_2), \tag{13}$$

$$X_1(t_k) = [x\ \dot{x}\ y\ \dot{y}\ z\ \dot{z}]', \tag{14}$$

$$X_2(t_k) = [x\ \dot{x}\ \ddot{x}\ y\ \dot{y}\ \ddot{y}\ z\ \dot{z}\ \ddot{z}]'. \tag{15}$$

It is worth noting here that

$$X_1(t_k) = M X_2(t_k), \tag{16}$$

where

$$\mathrm{M} = \begin{bmatrix} 1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,0 \end{bmatrix}. \tag{17}$$

Let $X(t_k)$ be the true target state whose actual dynamics are unknown. Let the true measurement model of the target state be given by

$$z(t_k) = H\,X(t_k) + V(t_k), \tag{18}$$

where $V(t_k)$ is a white Gaussian process noise;

$$V(t_k) \sim N(0, R_k). \tag{19}$$

The measurement models used by the local processors are given by

$$z_i(t_k) = H_i X_i(t_k) + V_i(t_k), \qquad \text{i=1, 2,} \tag{20}$$

where

$$H_1 = \begin{bmatrix} 1\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0 \end{bmatrix}, \tag{21}$$

$$H_2 = \begin{bmatrix} 1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,0\,0 \end{bmatrix}. \tag{22}$$

Note that, even though the measurement models given by Eqs. (20)-(22) are different, they must both produce the same numerical measurement of the target position as the one given by Eq. (18). This implies that the local models must be valid models in the sense that they produce the same measurement data as the true target model. Assuming that the measurement noise is modeled correctly in both the local and true target model, i.e., $V_i(t_k) = V(t_k)$, then the local models are valid if for any time t_k

$$H_1 X_1(t_k) = H_2 X_2(t_k) = H\,X(t_k). \tag{23}$$

A necessary and sufficient condition for Eq. (23) to hold is that there exist a mapping M_i between H and H_i, i=1, 2, such that [10]

$$H = H_i M_i. \tag{24}$$

3.3 ASSUMPTION COMMON TO BOTH THE MSSM AND SSMM PROCESSES

The analytical approach continues by assuming a functional form for the fused track. The form of the fused state is hypothesized as being a linear combination of the local state updates (at their respective times) and the fused state predicted to the time at which fusion is to occur:

$$X_f(t_k|t_k) = L_0 X_f(t_k|t_{k-1}) + L_1 X_1(t_1|t_1) + L_2 X_2(t_2|t_2) \ , \tag{25}$$

where t_{k-1} is the previous time at which the fused state was generated and $t_{k-1} \le t_i \le t_k$. In some cases, $L_0 = 0$, as will be seen later.

3.4 STEPS REQUIRED IN THE MSSM PROCESS

The MSSM process leads to the development of fusion algorithms which take asynchronous tracks (thus $t_{k-1} \le t_1 \prec t_2 \le t_k$) and obtains a fused state estimate at some desired time, t_k. The initial step is to substitute the Kalman update equations for tracks 1 and 2 and the prediction equation for the fused track into Eq. (25). Then the expected value of the error of the fused state is formed:

$$E[\tilde{X}_f(t_k|t_k)] = E[X_f(t_k|t_k) - X(t_k)] . \tag{26}$$

There are two requirements set. First, it is required that the fused state update be unbiased (the term in Eq. 26 is equal to zero) when the predicted fused state is unbiased. This leads to a relationship between L_0, L_1, and L_2. Then, the covariance of the fused state, $P_f(t_k|t_k)$, is formed and the relationship between L_0, L_1, and L_2 is used to reduce the covariance to a function of two variables; assume that it is a function of L_1 and L_2. Second, it is required that the trace of the covariance be minimized. In order to meet this requirement, the first derivative of the trace of $P_f(t_k|t_k)$ is taken with respect to L_1 and L_2; each of these two equations is then set equal to zero. This will result in two equations with two unknowns. These equations are solved for L_1 and L_2 and then they are used in the first relationship to obtain a solution to L_0. If $L_0 = 0$, then one uses the unbiased condition to obtain L_1 in terms of L_2, uses this relationship to reduce the number of unknowns in the equation for the covariance to one, takes the derivative of the covariance, sets it to zero, and obtains L_2. L_2 can then be use to find L_1.

One of the most important tasks in obtaining a track fusion formula from this technique is to obtain and use the correct expression for the expected value of the product of the predicted value of the i^{th} filter and the process noise of the j^{th} filter. It can be shown that (see [7], [8], and [9]):

$$\begin{aligned} E[\tilde{X}_i(t_k|t_{k-1})W'(t_j)] &= E\{[X_i(t_k|t_{k-1}) - X(t_k)]W'(t_j)\} \\ &= -Q_j^k = -\int_{t_j}^{t_k} \Phi(t_k, t)Gq(t)G'\Phi'(t_k, t)dt . \end{aligned} \tag{27}$$

3.5 STEPS REQUIRED IN THE SSMM PROCESS

The SSMM process leads to the development of MM filters that take two synchronous tracks ($t_1 = t_2 = t_k$) generated using measurement data from a single sensor and multiple filters and combines these tracks to obtain a fused estimate. There have been two methods developed for this process. The

first method assumes that each filter produces its own fused track and that this track is used as the input state on the next update cycle:

$$X_f^1(t_k|t_k) = L_{11}X_1(t_k|t_k) + L_{12}X_2(t_k|t_k), \tag{28}$$

$$X_f^2(t_k|t_k) = L_{21}X_1(t_k|t_k) + L_{22}X_2(t_k|t_k). \tag{29}$$

It is then proposed that there is another stage where the fused outputs of the two Kalman filters are fused:

$$X_f(t_k|t_k) = L_1 X_f^1(t_k|t_k) + L_2 X_f^2(t_k|t_k). \tag{30}$$

This is not the method that will be used to generate the ARMM algorithm; the second method, to be given next, is the one that will be used. A complete derivation of the results used in this first method can be found in [11]. In fact, the structure given by Eq. (30) was shown to be theoretically optimal where the switching coefficient are proportional to the local filter residues. As the noise level of sensor increases, the separation between switching coefficients becomes less obvious.

The second method is less complicated. Here, it is assumed that there is no feedback and one is attempting to solve for two coefficients:

$$X_f(t_k|t_k) = L_1 X_1(t_k|t_k) + L_2 X_2(t_k|t_k). \tag{31}$$

In other words, one starts by substituting the Kalman update equations for tracks 1 and 2 into Eq. (25) (with $L_0 = 0$). Next, the expected value of the error of the fused state, $E[\tilde{X}_f(t_k|t_k)]$, is formed (see Eq. (26)). In this case, it is not possible to determine if the expected values of the error for tracks 1 and 2 are unbiased. However, it is necessary to require that $E[\tilde{X}_f(t_k|t_k)]$ be independent of the true target state (which is unknown). This leads to a relationship between L_1 and L_2.

The requirement that the covariance be minimized as in the MSSM process is not used here. If one insists on keeping this requirement, it leads to values for L_1 and L_2 that are <u>not</u> responsive to changes in the target's dynamics. This is undesirable since a change in the target's dynamics should result in a change of the weightings (contributions) of the filter updates. In order to provide a method that adapts L_1 and L_2 to the dynamics of the target in a real time application, the residues of the two local processors, the one relationship between L_1 and L_2, and the assumption that

$$L_1 = \alpha M', \tag{32}$$

$$L_2 = diagonal(\beta, \beta, \gamma, \beta, \beta, \gamma, \beta, \beta, \gamma), \tag{33}$$

are used to generate their values. Here, $\gamma = 1$.

4.0 FUSION ALGORITHMS OBTAINED USING THE MSSM PATH

First, consider the following architecture where tracks from multiple sensors, that make their measurements at different times, are fused and the fused track is sent back to each sensor's Kalman filter

to be used as the input to the Kalman filter when determining the next update (Figure 1). One assumes for the solution the functional form given in Eq. (25). As shown in [7], the coefficients are given by:

$$L_0 = I - L_1\Phi(t_1,t_k) - L_2\Phi(t_2,t_k)\,, \tag{34}$$

$$L_1 = [I + \Phi^{-1}(t_1,t_k)M_{21}N^{-1}M_{12} - N^{-1}M_{12}]\Phi^{-1}(t_1,t_k)\,, \tag{35}$$

$$L_2 = [I - \Phi^{-1}(t_1,t_k)M_{21}]N^{-1}\,, \tag{36}$$

where

$$N = \Phi(t_2,t_k) - M_{12}\Phi^{-1}(t_1,t_k)M_{21}\,. \tag{37}$$

In Eqs. (35)-(37), the terms M_{12} and M_{21} are given by:

$$M_{12} = P_2(t_2|\,t_2)H_2'R_2^{-1}(t_k)H_2\left\{I - Q_2^k[P_f(t_k|\,t_{k-1}) - Q_1^k]^{-1}\right\}, \tag{38}$$

$$M_{21} = P_1(t_1|\,t_1)H_1'R_1^{-1}(t_k)H_1\left\{I - Q_1^k[P_f(t_k|\,t_{k-1}) - Q_2^k]^{-1}\right\}, \tag{39}$$

and

$$P_f(t_k|\,t_{k-1}) = \Phi(t_k,t_i)P_i(t_i|\,t_{k-1})\Phi'(t_k,t_i) + Q_i^k\,. \tag{40}$$

Also, the updated covariance of the fused track is given by

$$\begin{aligned} P_f(t_k|\,t_k) = {} & TP_f(t_k|\,t_{k-1})T' + TQ_1^k(L_1E_1)' + TQ_2^k(L_2E_2)' + (L_1E_1)Q_1^kT' + (L_2E_2)Q_2^kT' \\ & + (L_1E_1)Q_1^k(L_1E_1)' + (L_2E_2)Q_2^k(L_2E_2)' + L_1K_1R_1(t_k)(L_1K_1)' + L_2K_2R_2(t_k)(L_2K_2)', \end{aligned} \tag{41}$$

where

$$T = I - L_1E_1 - L_2E_2\,, \tag{42}$$

$$E_i = K_iH_i\Phi(t_i,t_k)\,. \tag{43}$$

On the other hand, it might be necessary to have an open loop architecture for generating a track fusion algorithm. In such an architecture, the track of each sensor is generated when a measurement in the desired time interval is taken and then they are fused to obtain the system track. There is no feedback (see Figure 2). Under the requirements that the solution be unbiased and and the covariance be minimized, the solution will only be MMSE. One proceeds in the following manner (see [9] for more details). The solution is assumed to have the form given by Eq. (25) with $L_0 = 0$. Following the format used in [7], one obtains

$$X_f(t_k|\,t_k) = L_1X_1(t_1|t_1) + L_2X_2(t_2|t_2)\,, \tag{44}$$

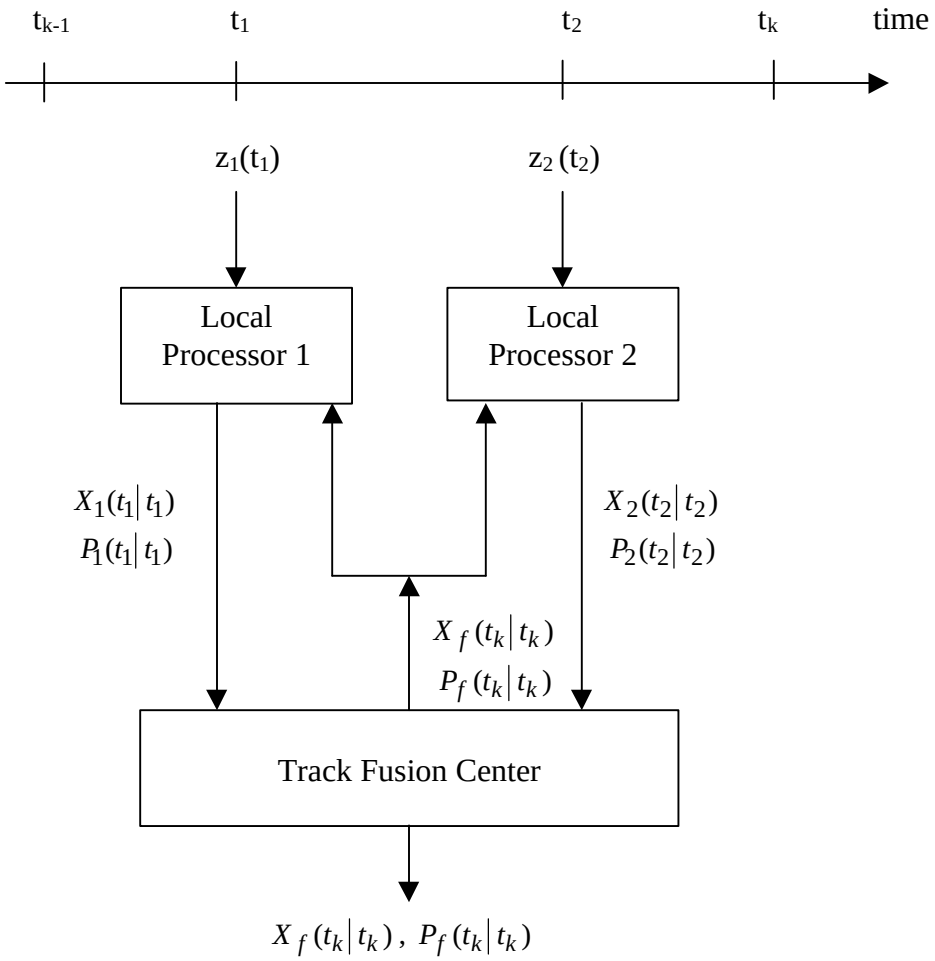


Figure 1. Asynchronous track fusion architecture with feed back.

$$\begin{aligned} P_f(t_k|t_k) = {} & L_1[M_{11} + \Phi(t_1,t_2)M_{22}\Phi(t_1,t_2)' - M_{12}\Phi(t_1,t_2)' - \Phi(t_1,t_2)M_{21}]L_1' \\ & - L_1[\Phi(t_1,t_2)M_{22} - M_{12}]\Phi(t_k,t_2)' - \Phi(t_k,t_2)[M_{22}\Phi(t_1,t_2)' - M_{21}]L_1' \\ & + \Phi(t_k,t_2)M_{22}\Phi(t_k,t_2)', \end{aligned} \tag{45}$$

where

$$L_1 = \Phi(t_k,t_2)[M_{22}\Phi(t_1,t_2)' - M_{21}][M_{11} + \Phi(t_1,t_2)M_{22}\Phi(t_1,t_2)' - M_{12}\Phi(t_1,t_2)' - \Phi(t_1,t_2)M_{21}]^{-1} \tag{46}$$

and

$$L_2 = [I - L_1\Phi(t_1,t_k)]\Phi(t_k,t_2)\,. \tag{47}$$

Here,

$$M_{11} = D_1P_1D_1' + D_1Q_1^kE_1' + E_1Q_1^kD_1' + E_1Q_1^kE_1' + K_1R_1K_1'\,, \tag{48}$$

$$M_{22} = D_2P_2D_2' + D_2Q_2^kE_2' + E_2Q_2^kD_2' + E_2Q_2^kE_2' + K_2R_2K_2'\,, \tag{49}$$

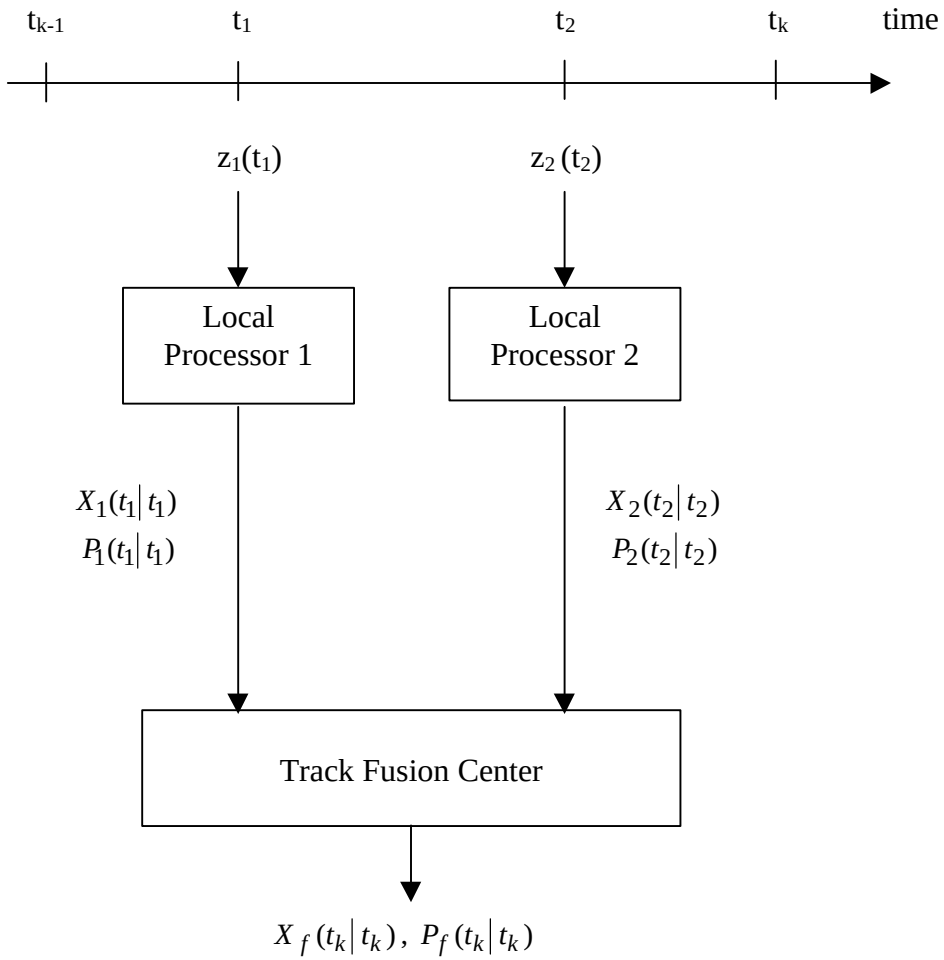


Figure 2. Open loop, asynchronous, track fusion architecture.

$$M_{12} = D_1 P_{12} D_2' + D_1 Q_2^k E_2' + E_1 Q_1^k D_2', \tag{50}$$

$$M_{21} = D_2 P_{21} D_1' + D_2 Q_1^k E_1' + E_2 Q_2^k D_1', \tag{51}$$

$$M_{21} = M_{12}', \tag{52}$$

where

$$P_f(t_k|t_k) = E\left[\tilde{X}_f(t_k|t_k)\tilde{X}_f'(t_k|t_k)\right], \tag{53}$$

$$P_i = P_i(t_k|t_{k-1}) = E\left[\tilde{X}_i(t_k|t_{k-1})\tilde{X}_i'(t_k|t_{k-1})\right], \tag{54}$$

$$P_{12} = P_{12}(t_k|t_{k-1}) = E\left[\tilde{X}_1(t_k|t_{k-1})\tilde{X}_2'(t_k|t_{k-1})\right], \tag{55}$$

$$P_{21} = P_{12}', \tag{56}$$

and

$$D_i = \Phi(t_i, t_k) - E_i . \tag{57}$$

Each of these two, asynchronous, track fusion algorithms have been shown, in the limit as $t_1, t_2 \to t_k$, to provide the same results as those obtained independently in earlier publications for sensors taking their measurements synchronously. Since the synchronous case is a special case of the asynchronous case, this lends credibility to these algorithms. This agreement has been demonstrated in [8] for the optimal, asynchronous, track fusion algorithm and in [9] for the MMSE, asynchronous, track fusion algorithm.

5.0 DERIVATION OF THE ARMM FILTER

Assume that the fusion center computes its track as

$$X_f(t_k \mid t_k) = L_1 X_1(t_k \mid t_k) + L_2 X_2(t_k \mid t_k) , \tag{58}$$

where $X_i(t_k \mid t_k)$ is the local state produced by LPj (local processor j), j=1, 2, $X_f(t_k \mid t_k)$ is the fused track at time k, and L_1 and L_2 are unknown switching coefficients (see Figure 3). Define

$$E_f(t_k \mid t_k) = X_f(t_k \mid t_k) - X(t_k) . \tag{59}$$

Using Eq. (58),

$$E_f(k \mid k) = L_1[X_1(k \mid k) - X_1(k)] + L_2[X_2(k \mid k) - X_2(k)] + L_1 X_1(k) + L_2 X_2(k) - X(k) . \tag{60}$$

Following the work in [11], one has

$$X_1(t_k) = MX(t_k) , \tag{61}$$

$$X_2(t_k) = X(t_k) . \tag{62}$$

It is worth noting that Eqs. (61) and (62) represent static relationships at time k and do not necessary hold between measurements. Using Eqs. (60)-(62), one obtains

$$E_f(t_k \mid t_k) = L_1 E_1(t_k \mid t_k) + L_2 E_2(t_k \mid t_k) + (L_1 M + L_2 - I) X(t_k) , \tag{63}$$

where $E_i(t_k \mid t_k)$ is the error of the state estimate of LPi at time t_k. To eliminate the dependence of $E_f(t_k \mid t_k)$ on X(t_k), one must have

$$L_1 M + L_2 = I . \tag{64}$$

Eq. (64) represents the first condition on the choice of the switching coefficients. In what follows, a practical structure will be imposed on these matrices. Namely, these switching coefficients will be restricted to:

$$L_1 = \mathsf{a} M' , \tag{65}$$

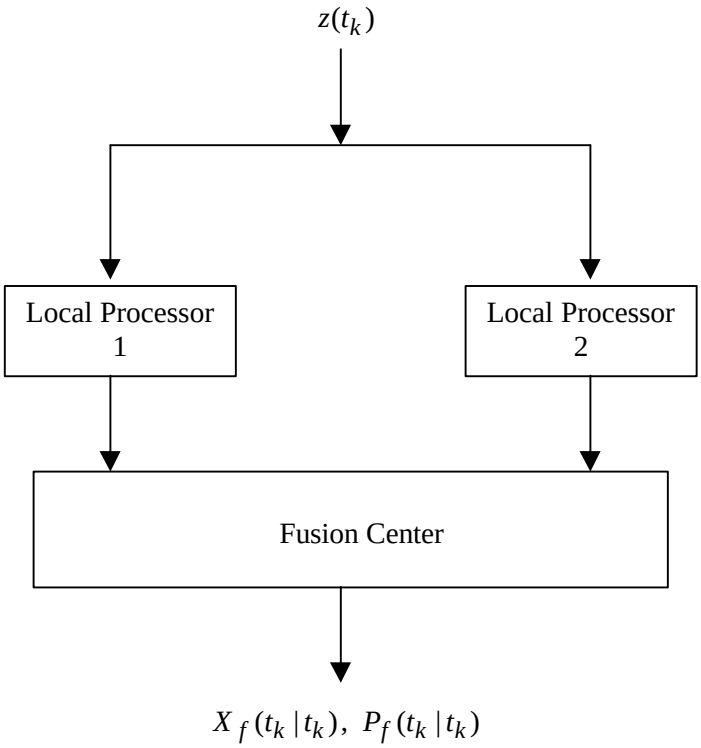


Figure 3. Open loop track fusion architecture

$$L_2 = diagonal(\beta, \beta, \gamma, \beta, \beta, \gamma, \beta, \beta, \gamma), \tag{66}$$

where $\gamma = 1$. Eqs. (64)-(66) imply

$$\alpha + \beta = 1. \tag{67}$$

To obtain another condition to determine the switching coefficients, with the above structure, using real time information , the residuals of the LPs will be used.

Let

$$r_1(t_k) = z(t_k) - H_1 \Phi_1(t_k, t_{k-1}) X_1(t_{k-1} \mid t_{k-1}), \tag{68}$$

$$r_2(t_k) = z(t_k) - H_2 \Phi_2(t_k, t_{k-1}) X_2(t_{k-1} \mid t_{k-1}). \tag{69}$$

It is proposed to choose α and β as follows:

$$\alpha = r_2(t_k)' r_2(t_k) / [r_1(t_k)' r_1(t_k) + r_2(t_k)' r_2(t_k)], \tag{70}$$

$$\beta = r_1(t_k)' r_1(t_k) / [r_1(t_k)' r_1(t_k) + r_2(t_k)' r_2(t_k)]. \tag{71}$$

The rationale for this choice is that if LP2 is doing poorly, then its residue will be large. The fusion center will then tilts toward LP1. Note that by using Eqs. (70) and (71), Eq. (67) is satisfied. Also, the

covariance of the fused state, $P_f(t_k \mid t_k)$, is the same as $P_f^2(t_k \mid t_k)$ found in Eq. (43) of [11] with L_{21} replaced by L_1.

6.0 SIMULATION RESULTS

In order to observe the effectiveness of the ARMM algorithm, it has been used to filter the data from a simulated target that has segments of time where it is flying at a constant velocity and other time segments where it is maneuvering. The profile of this target in the horizontal plane (X-Y plane) is given in Figure 4 and the vertical profile (motion along the z-axis) is given in Figure 5. Additionally, the velocities are given in Figures 6-8 and the accelerations are given in Figures 9-11.

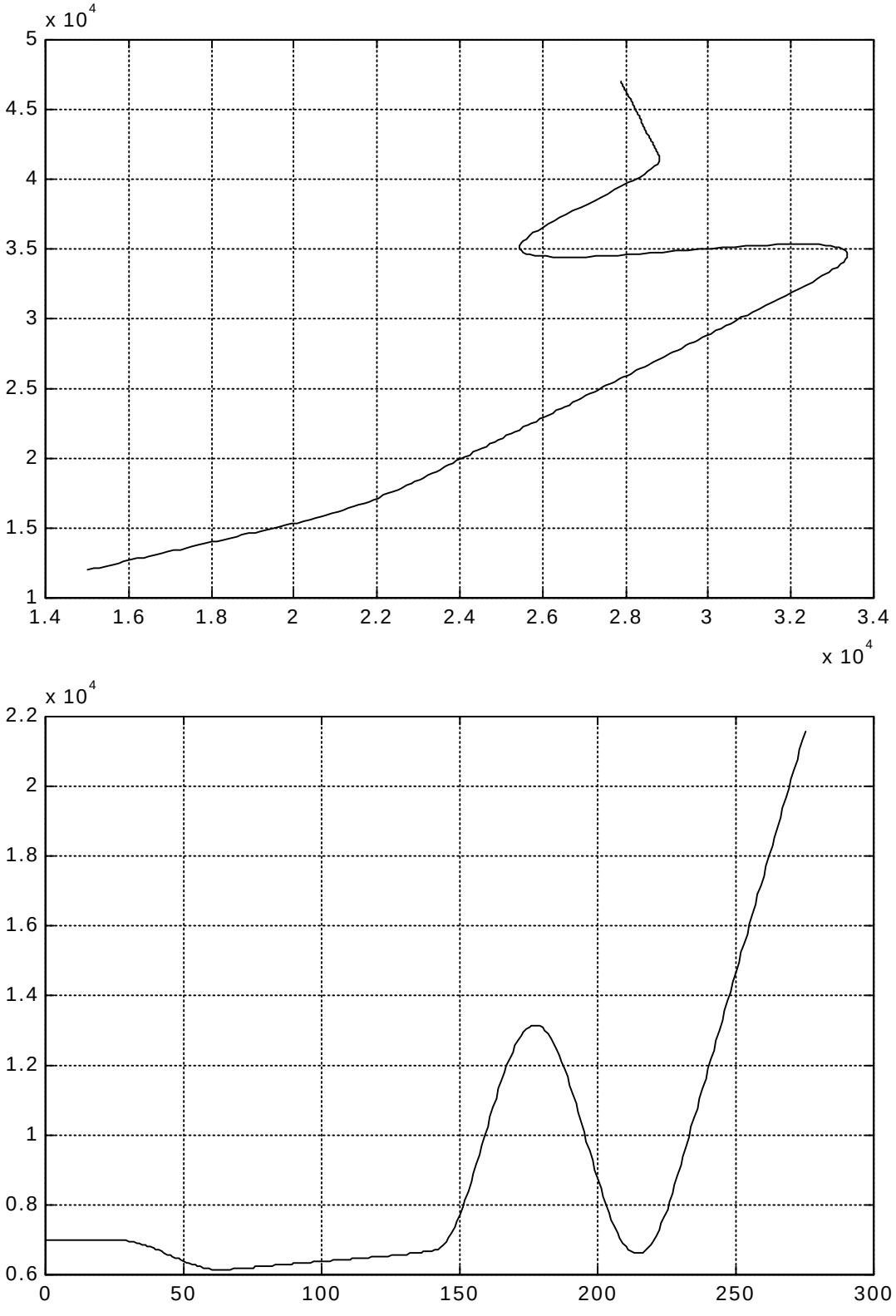
A simulated, three dimensional sensor was used to produce the measurement data and it's data rate was set at 1 Hz. The measurement noise covariance of the sensor is given by

$$R = \begin{bmatrix} r & r/20 & r/20 \\ r/20 & r & r/20 \\ r/20 & r/20 & r \end{bmatrix}, \tag{72}$$

where r is a parameter that has been given the values 10 m^2 (which represents a reasonably accurate sensor) and 100 m^2 (which is a relatively inaccurate sensor).

A CV and a CA filter were used in the ARMM algorithm. A process noise of q_1=100 was used for the CV filter and a process noise of q_2=200 was used in the CA filter. A Monte Carlo simulation of 50 runs was performed for each value of r. The switching coefficients, a and b, over the entire trajectory for the choice r=10 are given in Figure 12. There is good separation between a (denoted as 1 on the graph) and b (denoted as 2 on the graph) except in the region between t=30 seconds and t=70 seconds. In this region, the target underwent a weak maneuver. Because of the values chosen for q_1 and q_2, this weak maneuver appears to be a CV trajectory to the CV filter and a CA trajectory to the CA filter with equal likelihood. The RMS values of the speed over the trajectory (except for t=0 to t=50 seconds) for the CV filter (denoted as 1 on the graph), the CA filter (denoted as 2 on the graph), and the ARMM algorithm (denoted as 3 on the graph) are shown in Figure 13. It is seen that the RMS speed for the ARMM stays close to the CV filter's RMS value of the speed when the target is not maneuvering and stays near the CA filter's RMS value of the speed when it is maneuvering. The average velocity error is somewhere between 4m/s and 5 m/s. The RMS value of the position is not shown because of lack of space. It was felt that the error in the velocity is more critical when it comes to predicting the future position of the target in order to maintain track on the target than it's RMS position error at the time the track is updated.

For r=100, the switching coefficients over the entire trajectory are given in Figure 14. The switching coeffficients are slightly less well defined for this case than the r=10 case. (The a and b are identified by the same numbers used in Figure 12.) However, they still display good separation everywhere except in the region between t=30 seconds and t=70 seconds, for the same reason that they were not separated in this region when r=10. The RMS speed errors for the two filters and the ARMM are given in Figure 15 and are denoted by the same numbers used in Figure 13. The graphs in Figure 15 and Figure 13 look the same just the scale of the errors are larger in Figure 15. Here, the average RMS speed error for the ARMM alogrithm appears to be between 8m/s and 10 m/s.



Figures 4 and 5. The upper plot is figure 4 and it depicts the target motion in the X-Y plane or the target's horizontal profile (both axes are in meters). The lower plot depicts the target motion along the Z-axis or the target's vertical profile. The vertical axis is in meters and the horizontal axis is in seconds.

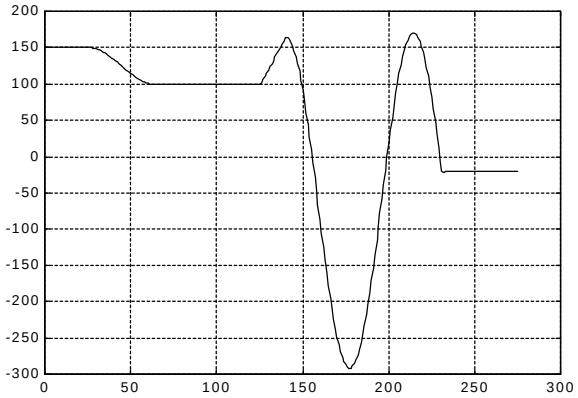


Figure 6. Plot of the target's x-velocity (vertical axis in meters/second) versus the time (horizontal axis in seconds). The start time of the trajectory is t=0 and the trajectory ends at approximately t=275 seconds.

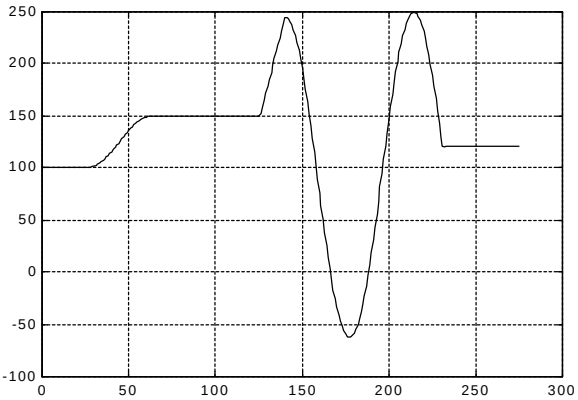


Figure 7. Plot of the target's y-velocity (vertical axis in meters/second) versus the time (horizontal axis in seconds). The start time of the trajectory is t=0 and the trajectory ends at approximately t=275 seconds.

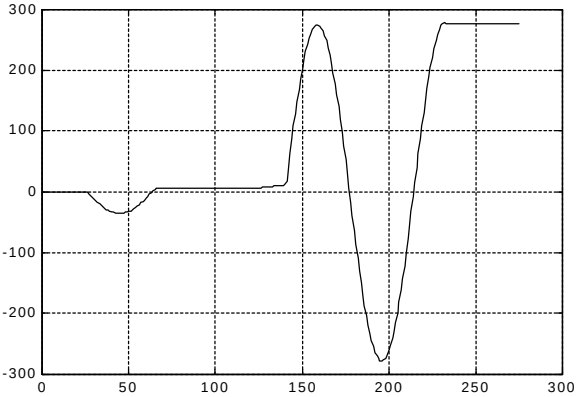


Figure 8. Plot of the target's z-velocity (vertical axis in meters/second) versus the time (horizontal axis in seconds). The start time of the trajectory is t=0 and the trajectory ends at approximately t=275 seconds.

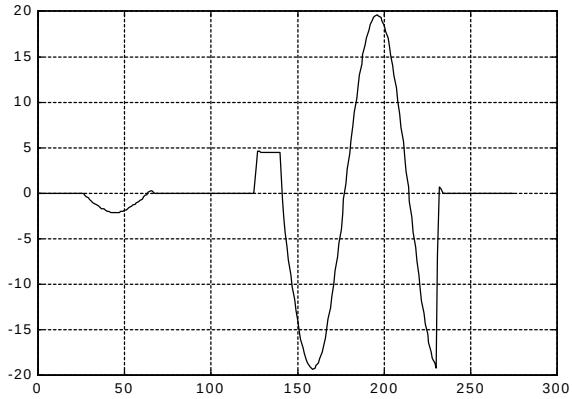


Figure 9. Plot of the target's x-acceleration (vertical axis in m/s^2) versus the time (horizontal axis in seconds). The start time of the trajectory is t=0 sec. and the trajectory ends at approximately t=275 sec.

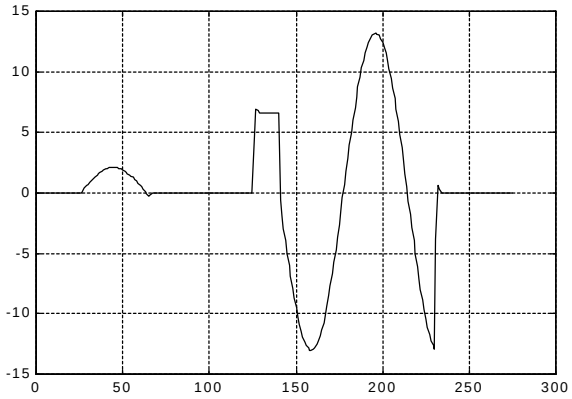


Figure 10. Plot of the target's y-acceleration (vertical axis in m/s^2) versus the time (horizontal axis in seconds). The start time of the trajectory is t=0 sec. and the trajectory ends at approximately t=275 sec.

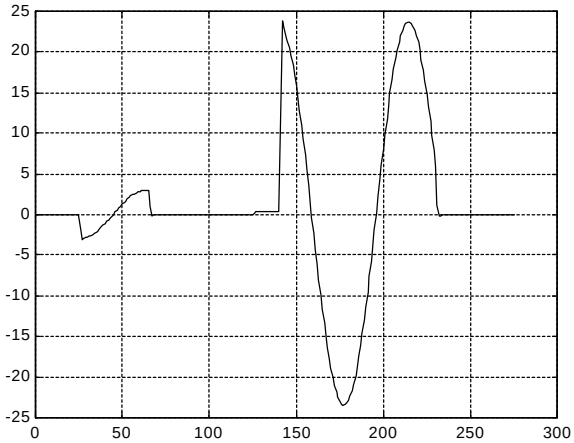


Figure 11. Plot of the target's z-acceleration (vertical axis in m/s^2) versus the time (horizontal axis in seconds). The start time of the trajectory is t=0 sec.and the trajectory ends at approximately t=275 sec.

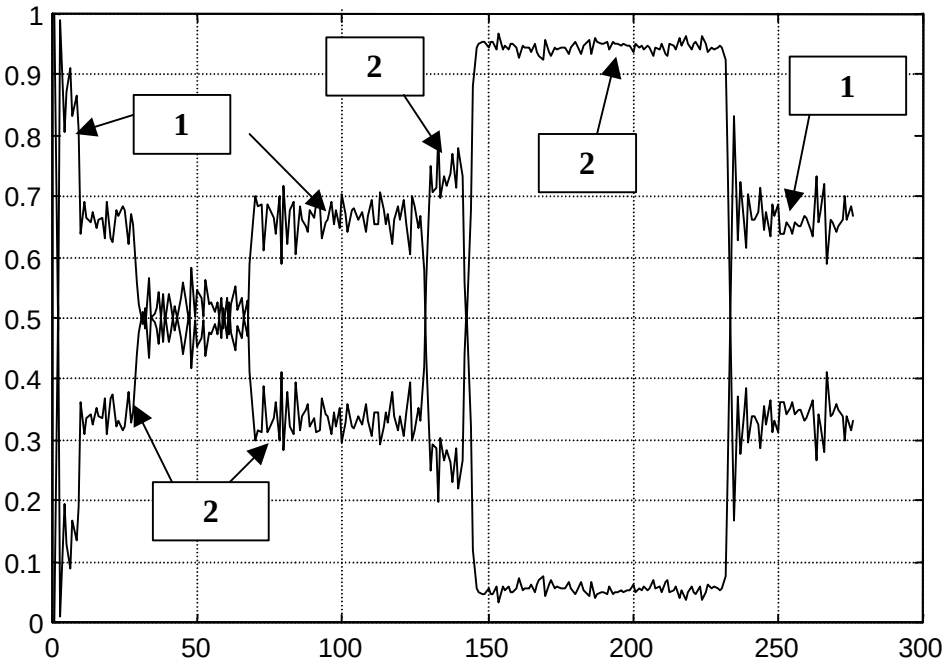


Figure 12. This figure shows the switching coefficients for the CV (1) and CA (2) filters over the trajectory for a reasonably accurate sensor (r=10). No delineation is made between the gains for the CV and CA filters from t=30 sec. to t=70 sec. because they are too close to each other.

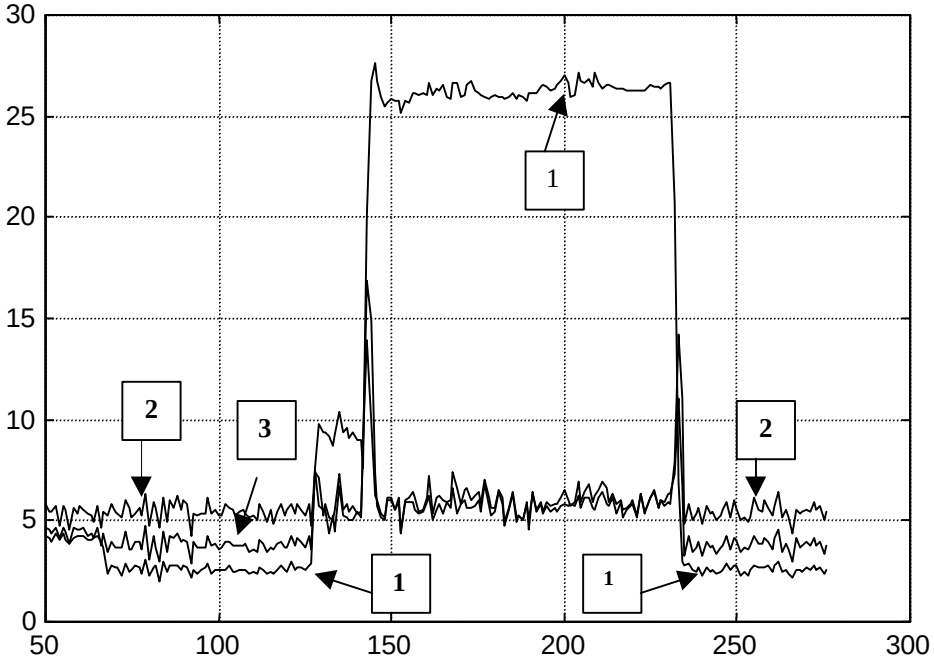


Figure 13. The RMS speed errors for the CV (1) and CA (2) filters and the ARMM filter algorithm (3) from t=50 sec. to t=275 sec. (r=10). The values for (3) lies between (1) and (2) when the target is slowly maneuvering (or not maneuvering at all) and is on top of (2) when the target is strongly maneuvering.

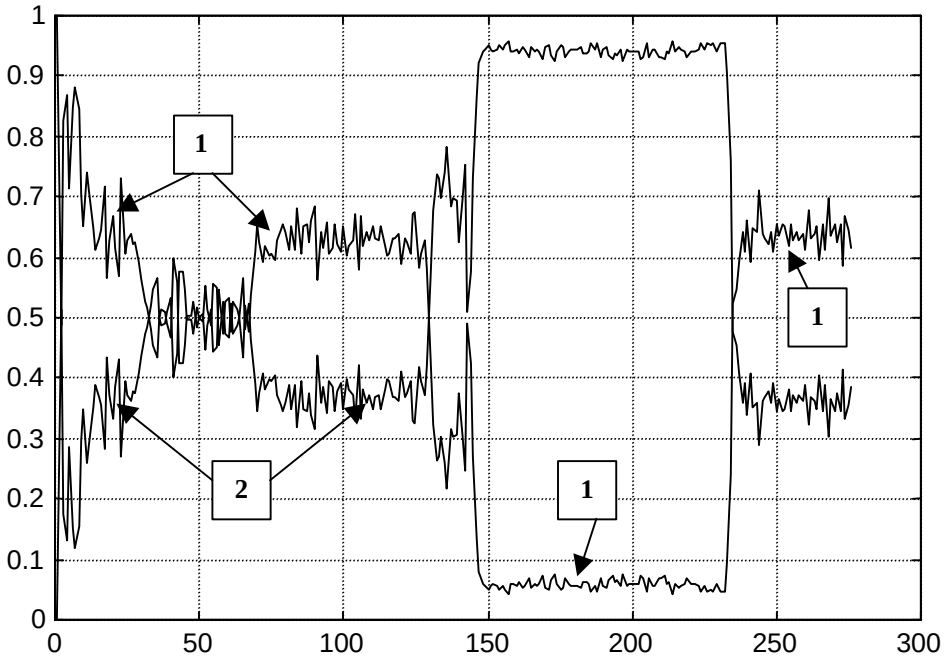


Figure 14. This figure shows the switching coefficients for the CV (1) and CA (2) filters over the trajectory for a more inaccurate sensor (r=100). No delineation is made between the gains for the CV and CA filters from t=30 sec. to t=70 sec. because they are too close to each other.

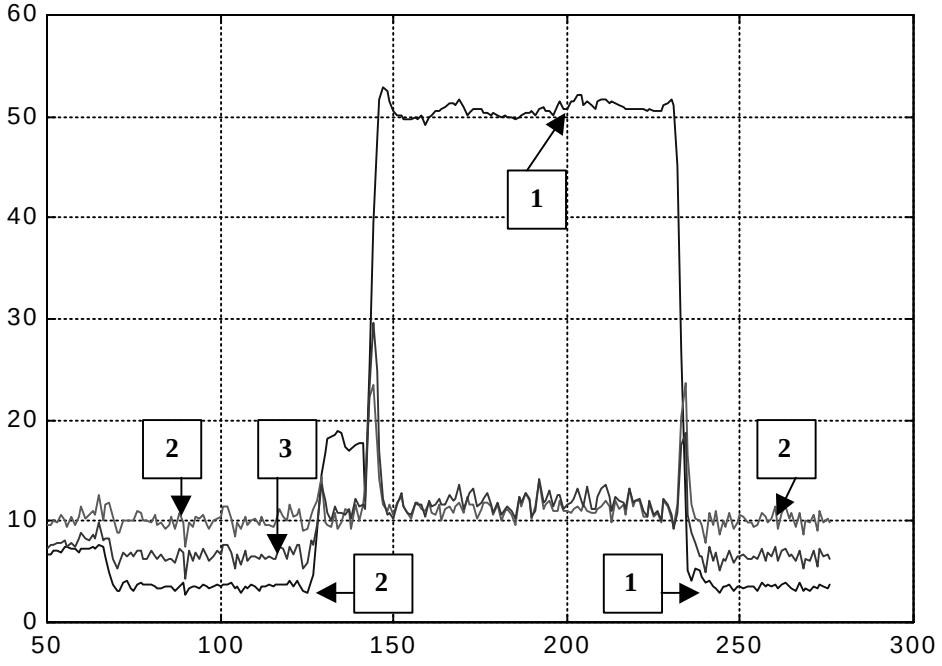


Figure 15. The RMS speed errors for the CV (1) and CA (2) filters and the ARMM filter algorithm (3) from t=50 sec. to t=275 sec. (r=100). The values for (3) lie between (1) and (2) when the target is slowly maneuvering (or not maneuvering at all) and is on top of (2) when the target is strongly maneuvering.

7.0 SUMMARY AND CONCLUSIONS

One of the important elements of this paper is the analytical approach discussed in section 3. From this analytical approach, it is possible to derive track fusion algorithms for sensors that provide tracks asynchronously and with communications delays. In particular, it is possible to derive an optimal track fusion algorithm for these sensors that can take asynchronously generated tracks and fuse them at any chosen point in time. Actually, this track fusion algorithm has some of the same properties as a Kalman filter. First, it uses tracks generated by filters using only a single model. Second, it is optimal when the target's dynamics match that model. Third, it uses its output state vector and covariance at t_{k-1} as input when determining the update at t_k. This analytic approach also has been used to derive a track fusion algorithm that is not optimal (only provides a MMSE solution) but will fuse the tracks from the same set of sensors under the same conditions.

The analytical approach also can be used to generate MM filters using data from a given sensor. The one derived in this paper is called the ARMM and provides a solution to the open loop architecture (no feedback required). The extensive simulation testing of this filter has not been performed yet. Therefore, the next step will be to compare how well it can track a target following a set of trajectories as compared to the IMM which appears to be the accepted standard at this time. Even though the investigations into it's capabilities are incomplete, the ARMM exhibits two important properties. First, the switching coefficients appear to be relatively unaffected by the sensors measurement inaccuracies. Second, the RMS error of the position (which was not shown in the paper) and of the speed for the fused state are close to those of the CV filter when the target is in the CV mode and close to those of the CA filter when the target is in the CA mode. This seems to indicate the successful on-line switching capability of the ARMM.

Beside being useful as a MM filter, the ARMM can also be used to produce a fused track from multiple sensors supplying local track updates asynchronously using filters having different dynamics models. This can be accomplished by using either of the two asynchronous track fusion algorithms presented in section 4. The optimal track fusion algorithm will be considered first because it has similarities to a Kalman filter. Since there is no feedback used in the other track fusion algorithm, its use in place of a Kalman filter is more tenuous.

Assume one has four filters tracking the same target, two with one model, say CV, and two with another model, say CA, and all of the measurements are the same dimensionally. The filters can be distributed among sensors in one of three ways: (1) the filters can each reside in a different sensor, (2) one of each model can reside in one sensor and each of the other two filters in a different sensors, or (3) two sensors can each have two filters, each of the two filters with a different model. Of course, under any of these three conditions some of the tracks may be synchronous. The optimal track fusion algorithm can be used once to fuse the tracks from the two filters with a CV model and then again to fuse the tracks from the two filters with the CA model. The two applications of the optimal track fusion algorithm would provide fused tracks at the same time. At this point, one can think of each of the two fused state vectors as the output of optimal, single model filters. They can be used as input to the ARMM algorithm in place of the input from two, single model, Kalman filters. The result is a fused track obtained from multiple asynchronous tracks generated with different dynamics models.

There is one important aspect that must be considered here. That is, what does one use for the filter residuals when calculating a and b in the ARMM algorithm? A reasonable and consistent choice is the following: predict the previous fused update for each model to the point of the last measurement taken by any one of the four sensors. That measurement is then used to form the residues as done in section 5 (Eqs. (68) and (69)).

The only difference in the use of the suboptimal track fusion algorithm instead of the optimal track fusion algorithm is that there is no information fed back to the local filters. On the positive side, this allows more variability in the local tracks used. For instance, if a MM algorithm such as an IMM algorithm is being used by all of the sensors, the tracks from the CV model filter in each IMM could be fused and then the tracks from the CA model filter in each IMM could be fused. Then the ARMM could be used to combine the fused CV track and the fused CA track. The local IMM filters could continue generating the local tracks to produce a local track picture. Of course, there are definitely concerns about the accuracy of this approach and they will be considered in the near future.

Finally, the use of an IMM structure in place of the ARMM structure is considered but only for the case when one is using the optimal track fusion algorithm in place of the Kalman filter. The IMM computes model likelihoods to be used in combining the output of the optimal track fusion algorithms for the CV and CA models and to compute the mixed state estimates (mixture of the previous output of the CV and CA model track fusion algorithms) used as input to the CV and CA track fusion algorithms. The main difference in the multisensor IMM structure and the multisensor ARMM structure is that: (1) the mixed fused state estimate is used as input to each optimal track fusion algorithm and is predicted to the time of the last measurement taken by any of the four sensors and (2) for each model, the residual covariance is used to normalize the residues which are then used to compute the model likelihoods.

REFERENCES

1. A. Jazwinsky, "Adaptive Filtering," *Automatica*, Vol. 5, May 1969.
2. Y. T. Chan, A. G. C. Hu, and J. B. Plant, "A Kalman Filter Based Tracking Scheme with Input Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-15, March 1979.
3. Y. Bar-Shalom and K. Birmiwal, "Variable Dimension Filter for Maneuvering Target Tracking," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-18, September 1982.
4. A. T. Alouani, P. Xia, W. D. Blair, and T. R. Rice, "On the Optimality of Two-Stage State Estimation in the presence of Random Bias," *IEEE Transaction on Automatic Control*, Vol. 38, No. 8, August 1993.
5. A. Chang, C. B. and M. Athans, "State Estimation for Discrete Systems with Switching Parameters," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-14, March 1978.
6. H. A P. Blom and Y. Bar-Shalom, "The Interacting Multiple Model Algorithm with Markovian Switching Coefficients," *IEEE Transactions on Automatic Control*, Vol. AC-33, August 1988.
7. A. T. Alouani and T. R. Rice, "On Optimal Asynchronous Track Fusion," *Proceedings of the First Australian Data Fusion Symposium*, Adelaide, Australia, November 21-22, 1996, pp. 147-152.
8. A. T. Alouani and T. R. Rice, "On Optimal Synchronous and Asynchronous Track Fusion," *Journal of Optical Engineering*, Vol. 37, No. 2, February 1998.
9. A. T. Alouani and T. R. Rice, "Asynchronous Fusion of Correlated Tracks," 12th International Symposium on Aerospace/Defense Sensing, Simulation, and Controls, (Proceedings of the Conference on Acquisition, Tracking, and Pointing XII), Orlando, Florida, 13-17 April 1998 (to appear)
10. A. S. Wilsky, *et al*, "Combining and Updating of Local Estimates and Regional Maps Along Sets of One-Dimensional Tracks," *IEEE Transactions on Automatic Control*, Vol. AC-27, August 1982.
11. T. R. Rice and A. T. Alouani, "Multiple Model Filtering," 12th International Symposium on Aerospace/Defense Sensing, Simulation, and Controls, (Proceedings of the Conference on Acquisition Tracking, and Pointing XII), Orlando, Florida, 13-17 April 1998 (to appear).